"Express Mail" mailing label number EV 304939613US

Date of Deposit: April 01, 2004

Attorney Docket No.15467US02

COMPRESSED STRUCTURE FOR SLICE GROUPS IN START CODE TABLE

RELATED APPLICATIONS

[0001] This application claims priority to Provisional Application for Patent, Serial No. 60/542,575, "Compressed Structure for Slice Groups in Start Code Table", filed February 5, 2004 by Thangaraj, et. al., and incorporated herein by reference.

FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0002] [Not Applicable]

[MICROFICHE/COPYRIGHT REFERENCE]

[0003] [Not Applicable]

BACKGROUND OF THE INVENTION

[0004] During the decoding of video data, the video data includes sets of data structures, such as transport packets, packetized elementary streams, elementary streams, group of pictures, pictures, slices, macroblocks, and blocks. Start codes indicate the starting points for groups of pictures, pictures, and slices.

[0005] The video includes parameters known as presentation time stamps (PTS) and decode time stamps (DTS) in the header of a packetized video elementary stream. The

PTS and DTS indicate the time that the next picture is to be decoded and presented for display, respectively. The video decoder compares the PTS and DTS against a program clock reference (PCR) to decode and present the picture for display.

[0006] A processor known as the video transport processor parses the headers of the transport packets, and the packetized elementary stream, and writes the video elementary stream into a memory known as a compressed data buffer. When the video transport processor writes the video elementary stream into the compressed data buffer, the video transport processor also maintains a table that records the address in the compressed data buffer to which each start code is written.

[0007] When the video decoder selects a data structure for decoding, the video decoder looks up the start code of the data structure in the start code table to determine the address storing the start code. The video decoder then fetches data starting from the address.

[0008] Because the PTS and DTS are in the header of the packetized video elementary sequence, the PTS, DTS, and PCR offset can be placed after the start code in the compressed data buffer for the first start code following a non-slice start code. Insertion of the PTS, DTS, and PCR offset in the compressed data buffer increases the number of bytes that need to be written. Each insertion after a non-slice start code could add as many as 17 bytes of data. When a transport packet contains a large number of non-slice start codes, insertion leads to compressed data going beyond 256 bytes. Accordingly, two DRAM access per packet are done in this case.

[0009] Further limitations and disadvantages of conventional and traditional approaches will become apparent to one of ordinary skill in the art through comparison of such systems with the present invention as set forth in the remainder of the present application with reference to the drawings.

BRIEF SUMMARY OF THE INVENTION

[0010] Presented herein is a compressed structure for writing slice group start codes into a start code table.

[0011] In one embodiment, there is presented a method for decoding video data. The method comprises writing one or more start codes to a start code table; and writing presentation time information to the start code table.

[0012] In another embodiment, there is presented a circuit for decoding video data. The circuit comprises a start code table and a video transport processor. The start code table stores start codes and comprises a plurality of data words. The video transport processor writes a plurality of start codes to a particular data word in the start code table.

[0013] In another embodiment, there is presented an article of manufacture. The article of manufacture comprises a machine readable medium. The machine readable medium stores a plurality of executable instructions. The plurality of executable instructions are for writing one or more start codes to a start code table; and for writing presentation time information to the start code table.

[0014] These and other features and advantages of the present invention may be appreciated from a review of the following detailed description of the present invention, along with the accompanying figures in which like reference numerals refer to like parts throughout.

BRIEF DESCRIPTION OF SEVERAL VIEWS OF THE DRAWINGS

[0015] **FIGURE 1** illustrates a block diagram of an exemplary Moving Picture Experts Group (MPEG) encoding process, in accordance with an embodiment of the present invention;

[0016] **FIGURE 2** is a block diagram of an exemplary decoder system in accordance with an embodiment of the present invention;

[0017] **FIGURE 3A** is a block diagram describing an exemplary base address entry in accordance with an embodiment of the present invention;

[0018] **FIGURE 3B** is a block diagram of an exemplary PTS/DTS entry in accordance with an embodiment of the present invention;

[0019] **FIGURE 3C** is a block diagram of a BTP Command and PCR Offset entry in accordance with an embodiment of the present invention;

[0020] **FIGURE 3D** is a block diagram of a start code entry in accordance with an embodiment of the present invention;

[0021] **FIGURES 4A and 4B** are a flow diagram for marking start codes in a transport packet in accordance with an embodiment of the present invention;

[0022] **FIGURE 5** is a block diagram of an exemplary start code table entry for a particular sequence of start codes;

[0023] **FIGURE 6** is a block diagram of another exemplary start code table entry for another particular sequence of start codes;

[0024] **FIGURE 7** is a block diagram of another exemplary start code table entry for another particular sequence of start codes;

[0025] **FIGURE 8** is a block diagram of another exemplary start code table entry for another particular sequence of start codes;

[0026] **FIGURE 9** is a block diagram of another exemplary start code table entry for another particular sequence of start codes;

[0027] **FIGURE 10** is a block diagram of another exemplary start code table entry for another particular sequence of start codes; and

[0028] **FIGURE 11** is a block diagram of another exemplary start code table entry for another particular sequence of start codes.

DETAILED DESCRIPTION OF THE INVENTION

[0029] **FIGURE 1** illustrates a block diagram of an exemplary Moving Picture Experts Group (MPEG) encoding process of video data 101, in accordance with an embodiment of the present invention. The video data 101 comprises a series of frames 103. Each frame 103 comprises two-dimensional grids of luminance Y, 105, chrominance red Cr, 107, and chrominance blue C_b, 109, pixels. The two-dimensional grids are divided into 8x8 blocks, where a group of four blocks or a 16x16 block 113 of luminance pixels Y is associated with a block 115 of chrominance red C_r, and a block 117 of chrominance blue C_b pixels. The block 113 of luminance pixels Y, along with its corresponding block 115 of chrominance red pixels C_r, and block 117 of chrominance blue pixels C_b form a data structure known as a macroblock 111. The macroblock 111 also includes additional parameters, including motion vectors, explained hereinafter. Each macroblock 111 represents image data in a 16x16 block area of the image.

[0030] The data in the macroblocks 111 is compressed in accordance with algorithms that take advantage of temporal and spatial redundancies. For example, in a motion picture, neighboring frames 103 usually have many similarities. Motion causes an increase in the differences between frames, the difference being between corresponding pixels of the frames, which necessitate utilizing large values for the transformation from one frame to another. The differences between the frames may be reduced using motion compensation, such that the transformation from frame to frame is minimized. The idea of motion

compensation is based on the fact that when an object moves across a screen, the object may appear in different positions in different frames, but the object itself does not change substantially in appearance, in the sense that the pixels comprising the object have very close values, if not the same, regardless of their position within the frame. Measuring and recording the motion as a vector can reduce the picture differences. The vector can be used during decoding to shift a macroblock 111 of one frame to the appropriate part of another frame, thus creating movement of the object. Hence, instead of encoding the new value for each pixel, a block of pixels can be grouped, and the motion vector, which determines the position of that block of pixels in another frame, is encoded.

[0031] Accordingly, most of the macroblocks 111 are compared to portions of other frames 103 (reference frames). When an appropriate (most similar, i.e. containing the same object(s)) portion (reference pixels) of a reference frame 103 is found, the difference between the portion of the reference frame 103 and the macroblock 111 are encoded. The difference is known as the prediction error. The location of the reference pixels in the reference frame 103 is recorded as a motion vector. The encoded difference and the motion vector form part of the data structure encoding the macroblock 111. In the MPEG-2 standard, the macroblocks 111 from one frame 103 (a predicted frame) are limited to prediction from portions of no more than two reference frames 103. It is noted that frames 103 used as a reference frame for a predicted frame 103 can be a predicted frame 103 from another reference frame 103.

[0032] The macroblocks 111 representing a frame are grouped into different slice groups 119. The slice group 119 includes the macroblocks 111, as well as additional parameters describing the slice group. Each of the slice groups 119 forming the frame form the data portion of a picture structure 121. The picture 121 includes the slice groups 119 as well as additional parameters that further define the picture 121.

[0033] The pictures are then grouped together as a group of pictures (GOP) 123. The GOP 123 also includes additional parameters further describing the GOP. Groups of pictures 123 are then stored, forming what is known as a video elementary stream (VES) 125. The VES 125 is then packetized to form a packetized elementary sequence 130.

[0034] The packetized elementary sequence 130 includes presentation time stamps and decode time stamps that indicate the time that the first following picture is to be decoded and presented for display. The packetized elementary stream is further packetized into fixed 192-byte (including a 4 byte header, and 188-bytes of data) packets, known as transport packets 135.

[0035] The transport packets 135 can be multiplexed with other transport packets carrying other content, such as another video elementary stream 125 or an audio elementary stream. The multiplexed transport packets 135 form what is known as a transport stream. The transport stream is transmitted over a communication medium for decoding and displaying.

[0036] Referring now to **FIGURE 2**, there is illustrated a block diagram describing an exemplary decoder system 200 in

accordance with an embodiment of the present invention. The decoder system 200 receives a transport stream 205 and stores the transport stream 205 in a transport stream presentation buffer 210. The transport stream presentation buffer 210 can comprise memory, such as synchronous dynamic random access memory (SD-RAM).

[0037] A transport processor 215 demultiplexes the transport stream 205 into constituent elementary streams. For example the transport stream can comprise any number of video and audio elementary stream constituents. Additionally, the transport processor 215 parses and processes the transport header information from the transport streams stored in the transport stream presentation buffer 210. The constituent audio elementary streams can be provided to an audio decoding section of the decoder system 200.

[0038] A video transport processor 218 parses the headers of the transport packets, and the packetized elementary stream. Additionally, the video transport processor 218 writes video elementary stream 125 to a compressed data buffer 220. As noted above, the video elementary stream 125 comprises a hierarchy of various structures, such as GOPs 123, pictures 121, slice groups 119, and macroblocks 111. The starting point of the foregoing is indicated in the video elementary stream 125 by what is known as a start code.

[0039] As the transport processor 218 writes the video elementary stream 125 to the compressed data buffer 220, the transport processor 215 also maintains a start code table 225. The start code table 225 comprises a memory storing records of start codes and the addresses in the

compressed data buffer 220 storing the start code and will be described in greater detail below. The memory can comprise, for example, a plurality of 16 byte data words, known as Gigantic Words (gwords).

[0040] A video decoder 230 decodes the video elementary stream 125 stored in the compressed data buffer 220. The video decoder 230 decodes the video elementary stream 125 on a picture-by-picture basis, and decodes the pictures on a slice-by-slice basis. As noted above, the PTS and DTS in the header of the packetized video elementary stream 130 indicate the time to decode and present a picture for display.

[0041] Because the PTS and DTS are in the header of the packetized video elementary sequence, the video transport processor 218 writes the PTS, DTS, and PCR offset in the start code table. The video decoder 230 uses the foregoing values to decode and present the pictures for display.

[0042] The format for writing entries in the start code table for start codes in a transport packet will now be described. The entries for start codes in a transport packet can include a base address entry, a PTS/DTS entry, a BTP and PCR Offset Entry, and a start code entry.

[0043] Base Address Entry - There is one base address entry per transport packet if there are one or more start codes in the transport packet. This entry contains the compressed data buffer address pointing to the first byte of the packet payload. The base address can be used along with the byte offset provided in the start code entry (described below) to calculate the byte address of the

start of each the first non-slice start code that follows a slice start code.

[0044] PTS/DTS - This entry is created for each new PTS/DTS that is received in the transport packet with an associated Non Slice start code.

[0045] BTP Command and PCR offset Entry - There is one BTP **[Inventor Question: what does this stand for?]** command and Program Clock Reference (PCR) Offset Entry per transport packet if a Non Slice start code is present in the transport packet. This entry can be made for every change in PCR offset and also whenever the PCR offset parameter, PCR_offset, validity changes (ie., Valid to invalid transition and Invalid to Valid transition). For simplicity, a PCR_OFFSET entry can be forced once for the first Non-slice detected in a packet. This way it ensures that the information on the transitions are not missed.

[0046] Start Code Entry - Each entry can hold the byte offset for maximum 7 start codes. 0xFF in the offset section of the start code indicates the end of the valid start codes.

[0047] Referring now to **FIGURE 3**, there is illustrated a block diagram describing an exemplary base address entry 300 in accordance with an embodiment of the present invention. The base address entry 300 consumes a single gword comprising 16 bytes, Byte 15...0. The base address entry 300 includes a code, code 0, code 1, indicating that the gword stores a base address entry 300 in the first two bytes, Byte 15, 14.

[0048] The byte address, CDB0 Addr., CDB1 Addr., CDB2 Addr., CDB3 Addr., in the compressed data buffer storing

the beginning of the transport packet payload follows the code, code 0, code 1, and is stored in the next four bytes, Bytes 13, 12, 11, 10.

[0049] The next two bytes, Bytes 9 and 8, store indicators, Error byte 0, Error byte 1, that indicate to the video decoder the errors detected in the Transport layer. The errors can include, but are not limited to, a continuity counter error, a CDB Overflow, or an SCT Buffer Overflow. The remaining bytes, Bytes 7...0, can be used for other purposes, or not be used at all.

[0050] Referring now to **FIGURE 3B**, there is illustrated a block diagram describing an exemplary PTS/DTS Entry 325. If the transport packet includes a PTS and DTS value and an associated non-slice start code, the video transport processor writes a PTS/DTS entry 325 into the compressed data buffer. The PTS/DTS Entry 325 consumes a gword comprising 16 bytes, Bytes 15...0. The first two bytes, Bytes 15, 14, store a code, code 0, code 1, indicating that the gword stores a PTS/DTS entry.

[0051] The next two bytes, Bytes 13 and 12, store flag bytes, Flag byte 0, Flag byte 1, to indicate the validity of the PTS and DTS values in the entry. The next five bytes, Bytes 11...7 store the PTS value in the transport packet, PTS0...PTS4. The next five bytes, Bytes 6...2 store the DTS value in the packet, DTS0...DTS4.

[0052] Referring now to **FIGURE 3C**, there is illustrated a block diagram describing an exemplary BTP command and PCR offset entry 350. One BTP Command and PCR Offset Entry 350 is written for a transport packet, if the transport packet includes at least one non-slice start code.

[0053] The BTP Command and PCR Offset Entry 350 consumes a gword comprising 16 bytes, Bytes 15...0. The first two bytes, Bytes 15, 14, store a code, Code 0, Code 1, indicating that the gword stores as BTP Command and PCR Offset Entry 350. The next two bytes, Bytes 13, 12 store flag bytes, Flag byte 0, Flag byte 1. This field is used to indicate the validity of the BTP and PCR_OFFSET values in the entry. The following eight bytes, Bytes 11...4, can store a BTP command, BTP Cmd0...BTP Cmd7, extracted from the Adaptation field of the BTP packet can be placed into this field. The last four bytes, Bytes 3...0 can store a PCR offset, PCR0 Offset...PCR3 Offset, calculated by the transport processor.

[0054] Referring now to **FIGURE 3D**, there is illustrated a block diagram describing an exemplary start code entry 375. The start code entry stores 375 can store a non-slice start code, and each slice start code following the non-slice start code, until the next non-slice start code.

[0055] The start code entry 375 consumes a gword comprising 16 bytes, Bytes 15...0. The first two bytes, Bytes 15, 14, store a code, code 0, code 1, indicating that the gword stores a start code entry 375. The remaining bytes, Bytes 13...0 store start code values, SC value, and byte offsets, Byte Offset, in alternating order. As many as seven start code values, SC0 Value...SC6 Value, and byte offsets, Byte Offset 0...Byte Offset 6, can be stored.

[0056] The start code values, SC0 Value...SC6 Value are valid if the corresponding byte offset, Byte Offset 0...Byte Offset 6 does not store a 0xFF value. The byte offset, Byte Offset 0...Byte Offset 6 is the offset in bytes from the Base address to the address storing the

start code, SC0 Value...SC6 Value in the compressed data buffer. When the byte offset = 0xFF it indicates the SC value associated with this offset is invalid and the SC value in the previous pair is the last valid Start code in this entry. The video decoder can halt parsing beyond the Byte offset with 0xFF.

[0057] Referring now to **FIGURES 4A** and **4B**, there is illustrated a flow diagram describing the operation of the video transport processor 218 in accordance with an embodiment of the present invention. Starting with **FIGURE 4A**, at 405 the video transport processor 218 writes a CDB entry 300 for the transport packet into the next gword of the start code table 225.

[0058] At 410, the video transport processor 218 selects the first start code in the transport packet. At 415, the video transport processor 218 determines whether the first start code in the transport packet is a slice start code, or a non-slice start code.

[0059] If the first start code is a slice start code, the video transport processor 218 writes the slice start code, and the offset to the next gword at 420. At 425, a determination is made whether the next start code is a slice start code or a non-slice start code. Each subsequent start code and corresponding offset are written in the subsequent bytes of the current gword at 430 in the start code table 225, until a non-slice start code is found during 425. When a non-slice start code is found during 425, the video transport processor 218 writes another entry at 435 in the next bytes of the gword, including a dummy start code and the offset 0xFF, indicating that the dummy

start code is not valid, and that there are no more start codes in the gword.

[0060] Where the first start code is a non-slice start code during 415 or after 435, the video transport processor 218 makes a determination at 440 whether the non-slice start code is associated with a PTS/DTS. If the non-slice start code is associated with a PTS/DTS, then the video transport processor 218 writes a PTS/DTS entry in the next gword in the start code table 225 at 445. If the non-slice start code is not associated with a PTS/DTS, 445 is bypassed.

[0061] At 450, the video transport processor 218 determines whether there is a BTP Command and PCR offset associated with the non-slice start code. If there is a BTP Command and PCR offset associated with the non-slice start code at 450, the video transport processor 218 writes (455) a BTP command and PCR offset entry into the next gword of the start code table 225. If there is not a BTP Command and PCR offset associated with the non-slice start code at 450, 455 is bypassed. At 460, the video transport processor 218 writes the non-slice start code and offset to the next gword in the start code table 225.

[0062] Moving to **FIGURE 4B**, at 465, a determination is made whether there is another start code in the transport packet. If there is another start code in the transport packet at 465, a determination is made at 470 whether the next start code is a slice start code or a non-slice start code. If the next start code is a slice start code at 470, the video transport processor 218 writes the start code and an offset in the next bytes of the gword at 475. The video transport processor returns to 465.

[0063] If at 470, the next start code is a non-slice start code, the video transport processor 218 determines at 480, whether there is PTS/DTS value associated with the non-slice start code. If at 480, the video transport processor 218 determines there is a PTS/DTS value associated with the non-slice start code, the video transport processor 218 at 485 writes a PTS/DTS entry in the next gword of the start code table 225, and writes the non-slice start code in the next gword of the start code table 225 at 490.

[0064] If at 480, the video transport processor 218 determines there is not a PTS/DTS value associated with the non-slice start code, the video transport processor 218 writes at 495, the non-slice start code and offset in the next bytes of the gword. The foregoing, 465-490 are repeated until there are no more start codes in the transport packet at 465.

[0065] The foregoing is now further explained with the following illustrative examples of start code table 225 entries for certain transport packets.

[0066] Referring now to **FIGURE 5**, there is illustrated a block diagram describing the start code table 225 entry for a transport packet, wherein the start code packet includes the following start codes: Non-Slice Start Code (NSL)1 Slice Start Code (SL)1 SL2 SL3 with a new PTS/DTS, BTP Command, and PCR offset.

[0067] The start code table 225 entry includes a base address entry 300 in one gword, Gword0, a PTS/DTS entry 325 in the next gword, gword1, a BTP Command and PCR Offset Entry 350 in the next gword 2, and a start code entry 375

in the next gword, gword3. The start code entry 375 includes the start codes and offsets for NSL1, SL1, SL2, and SL3. The start code entry 375 also includes a dummy start code, XX, with the corresponding offset set to 0xFF to indicate that there are no other start codes in gword3.

[0068] Referring now to **FIGURE 6**, there is illustrated a block diagram of the start code table entry for a transport packet with the start codes: NSL1 SL1 SL2 SL3 and no new PTS/DTS, BTP Command, PCR offset. The start code table entry includes a base address entry 300 and a start code entry 375. The start code entry 375 includes the start codes and offsets for NSL1, SL1, SL2, and SL3. The start code entry 375 also includes a dummy start code, XX, with the corresponding offset set to 0xFF to indicate that there are no other start codes in gword1.

[0069] Referring now to **FIGURE 7**, there is illustrated a block diagram for a start code table entry for a transport packet with the start codes: SL1, SL2, SL3 and a new PTS/DTS, BTP Command, and PCR offset.

[0070] The start code table entry includes a base address entry 300 and a start code entry 375. The start code entry 375 includes the start codes and offsets for SL1, SL2, and SL3. The start code entry 375 also includes a dummy start code, XX, with the corresponding offset set to 0xFF to indicate that there are no other start codes in gword1.

[0071] Although the transport packet includes a new PTS/DTS, BTP Command, and PCR offset, the video transport processor 218 does not write a PTS/DTS entry or BTP Command

and PCR offset entry because the transport packet does not include a non-slice start code.

[0072] Referring now to **FIGURE 8**, there is illustrated a block diagram of an exemplary start code table entry for a transport packet with the start codes NSL1 SL1 NSL2 SL2, a new BTP Command, and PCR offset, and a new PTS/DTS associated with NSL1.

[0073] The start code table 225 entry includes a base address entry 300 in one gword, Gword0, a PTS/DTS entry 325 in the next gword, gword1, a BTP Command and PCR Offset Entry 350 in the next gword 2, and a start code entry 375 in the next gword, gword3. The start code entry 375 includes the start codes and offsets for NSL1, SL1, SL2, and SL3. The start code entry 375 also includes a dummy start code, XX, with the corresponding offset set to 0xFF to indicate that there are no other start codes in gword3.

[0074] Referring now to **FIGURE 9**, there is illustrated a block diagram of a start code table entry for a transport packet the includes start codes: NSL1 SL1 NSL2 SL2 , a new PCR offset, and BTP Command, and a new PTS/DTS associated with NSL2.

[0075] The start code table entry includes a base address entry 300 in the first gword, gword0, a BTP command and PCR Offset entry 350 in the next gword, gword1, a first start code entry 375 in the next gword, gword2, a PTS/DTS entry 325 in the next gword, gword3, and a second start code entry 375 in the next gword, gword4.

[0076] The BTP command and PCR Offset entry 350 are written before the first non-slice start code, NSL1. Since the PTS/DTS information is associated with NSL2, the

PTS/DTS entry 325 is written immediately before NSL2, and after SL1. In the first start code entry 375, a dummy start code, XX, is written corresponding to an offset of 0xFF, indicating that there are no more start codes in the first start code entry 375 in gword2. Similarly, in the second start code entry 375, a dummy start code, XX, is written corresponding to an offset of 0xFF, indicating that there are no more start codes in the first start code entry 375 in gword4.

[0077] Referring now to **FIGURE 10**, there is illustrated a block diagram describing the start code table entry for a transport packet including the start codes: NSL1 SL1 NSL2 SL2 NSL3 SL3 NSL4 SL4 NSL5 SL5 NSL6 and SL6, a new BTP Command, PCR offset, and new PTS/DTS associated with NSL1. The start code table entry includes a base address entry 300, a PTS/DTS entry 325, a BTP command and PCR Offset entry 350, a first start code entry 375, and a second start code entry 375.

[0078] Because the first start code is a non-slice start code, NSL1, and is associated with the PTS/DTS information, the a PTS/DTS entry 325, a BTP command and PCR Offset entry 350 are written prior the NSL1 in the first start code entry 375. The start codes NSL2, SL2, NSL3, SL3, and NSL4 are written in the first start code entry 375 following NSL1 in gword1. The remaining start codes, SL4, NSL5, SL5, NSL6, and SL6 are written in the second start code entry 375 in gword4, because there is no room for the start codes in gword3.

[0079] Referring now to **FIGURE 11**, there is illustrated a block diagram describing an exemplary start code table entry for a transport packet with start codes: SL1, SL2,

SL3, NSL1, and SL4, with a new BTP Command, a PCR offset, and a new PTS/DTS associated with NSL1.

[0080] The start code table entry includes a Base Address entry 300, a first start code entry 375, a PTS/DTS entry 325, a BTP Command and PCR Offset entry 350, and a second start code entry 375.

[0081] Because the non-slice start code NSL1 is associated with the PTS/DTS and is the first non-slice start code, the PTS/DTS entry 325 and the BTP command and PCR Offset entry 350 follow the first start code entry 375 with the start codes and offsets for SL1, SL2, and SL3. The BTP command and PCR Offset entry 350 are followed by the second start code entry 375 with the start codes and offsets for NSL1 and SL4.

[0082] One embodiment of the present invention may be implemented as a board level product, as a single chip, application specific integrated circuit (ASIC), or with varying levels integrated on a single chip with other portions of the system as separate components. The degree of integration of the system will primarily be determined by speed and cost considerations. Because of the sophisticated nature of modern processors, it is possible to utilize a commercially available processor, which may be implemented external to an ASIC implementation of the present system. Alternatively, if the processor is available as an ASIC core or logic block, then the commercially available processor can be implemented as part of an ASIC device with various functions implemented as firmware. For example, the flow chart of FIGURES 4A and 4B can be implemented as a set of executable instructions in the firmware.

[0083] While the present invention has been described with reference to certain embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted without departing from the scope of the present invention. In addition, many modifications may be made to adapt a particular situation or material to the teachings of the present invention without departing from its scope. Therefore, it is intended that the present invention not be limited to the particular embodiment disclosed, but that the present invention will include all embodiments falling within the scope of the appended claims.